



# Agentes de IA: Uma Introdução Abrangente com LangChain

Explorando a Fronteira da Inteligência Artificial Autônoma

# Introdução aos Agentes de IA

*"Agentes de IA são sistemas de software que usam a IA para alcançar objetivos e concluir tarefas em nome dos usuários. Eles demonstram raciocínio, planejamento e memória, com autonomia para tomar decisões, aprender e se adaptar."*

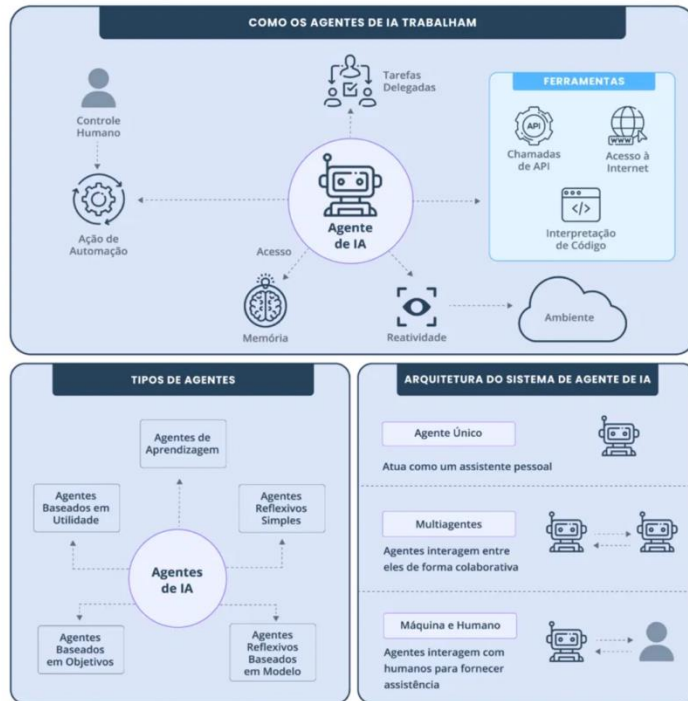
## Componentes Chave:

- 👁️ **Percepção:** Sensores para perceber o ambiente (e.g., texto, imagem, áudio).
- 🧠 **Raciocínio:** Motor de inferência (geralmente um LLM) para tomar decisões.
- ⚙️ **Ação:** Atuadores para interagir com o ambiente (e.g., executar código, chamar APIs).

## Objetivo:

Agir de forma autônoma para atingir metas pré-definidas.

### O QUE SÃO AGENTES DE IA



# A Evolução dos Agentes

## Agentes Reativos Simples

Agem com base na percepção atual (condição-ação).

## Agentes Baseados em Modelos

Mantêm um estado interno para entender o mundo.

## Agentes Baseados em Objetivos

Agem para atingir metas específicas.

## Agentes Baseados em Utilidade

Tentam maximizar uma medida de desempenho (utilidade).

## Agentes de Aprendizagem

Melhoram seu desempenho com o tempo através da experiência.

## Agentes com LLMs (Estado da Arte)



# Desafios no Uso de Agentes de IA

## Complexidade e Risco Ético

Dificuldade em lidar com a sutileza das emoções humanas e interações sociais complexas. Falta de um "compasso moral" para tomar decisões em situações eticamente sensíveis (e.g., saúde, aplicação da lei).

## Ambientes Imprevisíveis

Dificuldade em operar em ambientes físicos dinâmicos que exigem adaptação em tempo real e habilidades motoras finas.

## Recursos Computacionais

O desenvolvimento e a implantação de agentes sofisticados podem ser computacionalmente caros e intensivos em recursos.

## Segurança

A autonomia dos agentes representa um desafio crescente para a cibersegurança, pois podem ser explorados para fins maliciosos.





# LangChain e a Revolução dos Agentes

## O que é LangChain?

Um framework de código aberto para desenvolver aplicações alimentadas por modelos de linguagem (LLMs), facilitando a criação de agentes inteligentes e sistemas baseados em LLMs.

## Como o LangChain Ajuda?

-  **Abstração:** Fornece componentes modulares e reutilizáveis (Chains, Tools, etc.).
- Orquestração:** Facilita a criação de fluxos complexos que combinam LLMs com outras fontes de dados e APIs.
- Agentes:** Oferece uma estrutura de alto nível para construir agentes que usam LLMs para decidir quais ações tomar.
-  **Ferramentas:** Disponibiliza um ecossistema rico de ferramentas que os agentes podem utilizar para interagir com o mundo.



# Estrutura de um Agente LangChain

## 1. LLM (Large Language Model)

O cérebro do agente, responsável pelo raciocínio e tomada de decisão.

## 2. Tools (Ferramentas)

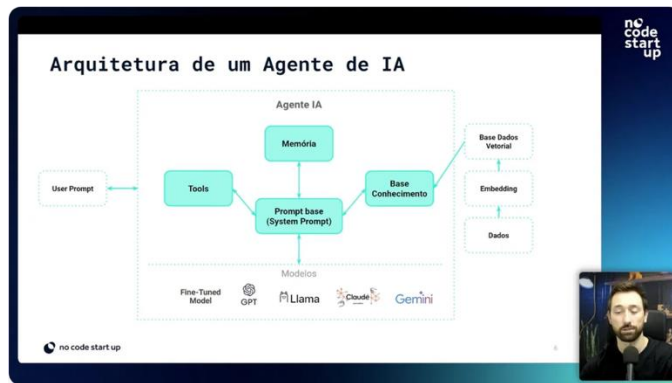
Funções que o agente pode executar para interagir com o mundo exterior.

## 3. Agent Executor

O ambiente de execução que orquestra a interação entre o LLM e as ferramentas.

## 4. Prompt Template

Um modelo que guia o LLM em seu processo de raciocínio.



# O Ciclo de Funcionamento (ReAct)

## ReAct (Reasoning and Acting)

Um paradigma popular para o funcionamento de agentes que combina raciocínio e ação em um ciclo iterativo.

### O Ciclo:

#### 1 Observação (Observation)

O agente recebe uma entrada do usuário ou percebe o estado atual do ambiente.

#### 2 Pensamento (Thought)

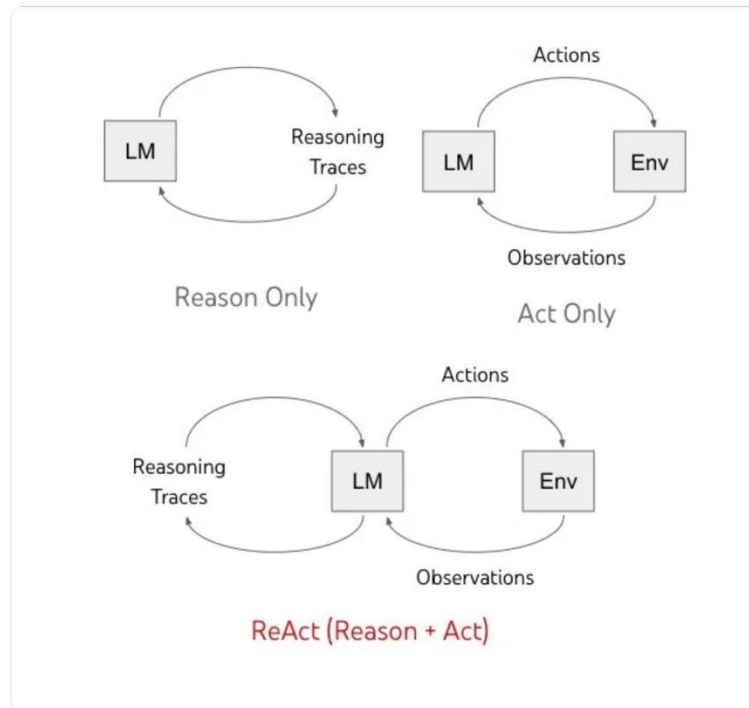
O LLM analisa a observação, decide o que fazer a seguir e qual ferramenta usar.

#### 3 Ação (Action)

O agente executa a ferramenta escolhida com os parâmetros definidos pelo LLM.

#### 4 Nova Observação

O agente recebe o resultado da ação, que se torna a nova observação para o próximo ciclo.



# O Poder das Ferramentas (Tools)

*"Ferramentas são abstrações que associam uma função Python a um esquema que define o nome, a descrição e os argumentos esperados da função."*

## Importância:

**Estendem as Capacidades do LLM:** Permitem que o LLM, que por si só apenas gera texto, interaja com o mundo real.

**Acesso a Informações Externas:** Busca de dados em tempo real (e.g., Google Search, APIs de notícias).

**Execução de Código:** Realização de cálculos, manipulação de dados (e.g., Python REPL, Shell).

**Interação com Sistemas:** Conexão com bancos de dados, APIs de serviços, etc.



### Ferramentas de Informação

Permitem que o agente busque e recupere informações de fontes externas, como motores de busca, bases de conhecimento e APIs.



### Ferramentas de Código

Possibilitam que o agente execute código, realize cálculos complexos e manipule dados usando linguagens como Python.

### Ferramentas de Integração

Conectam o agente a serviços externos, como APIs de terceiros, bancos de dados e sistemas de armazenamento.



### Ferramentas de Sistema

Permitem que o agente interaja com o sistema operacional, execute comandos de shell e gerencie arquivos.



# Criando uma Ferramenta em LangChain

## Simplicidade com o Decorator @tool

```
from langchain_core.tools import tool

@tool
def search_academic_papers ( query: str )-> list :
    """Busca por artigos acadêmicos sobre um determinado tópico."""
    # Lógica para chamar uma API de busca acadêmica
    # (e.g., arXiv, Google Scholar)
    return api.search(query)
```

## Como usar a ferramenta:

```
# Invocação direta
result = search_academic_papers.invoke({
    "query" : "agentes de IA"
})
```

Dica:

## O que o @tool faz?

### Inferência automática:

**Nome:** Inferido do nome da função ( `search_academic_papers` )

**Descrição:** Extraída da docstring da função

**Argumentos:** Inferidos das anotações de tipo ( `query: str` )

### Inspeção da ferramenta:

```
print (search_academic_papers.name)
# search_academic_papers

print (search_academic_papers.description)
# Busca por artigos acadêmicos sobre um determinado tópico.

print (search_academic_papers.args)
# {
#   'type': 'object',
#   'properties': {'query': {'type': 'string'}},
#   'required': ['query']
# }
```

# Arquiteturas de Agentes

## Agente Único (Single Agent)

Componentes de um Agente Único



### Exemplo:

Um agente de pesquisa que pode usar ferramentas de busca na web e leitura de artigos para responder a uma pergunta.

### Frameworks:

## Multi-Agente (Multi-Agent)

Componentes de um Sistema Multi-Agente



### Exemplo:

Um time de desenvolvimento de software com um agente "Gerente de Projeto", um "Programador" e um "Testador", cada um com suas próprias ferramentas e responsabilidades.

### Frameworks:

# Conclusão e Futuro dos Agentes

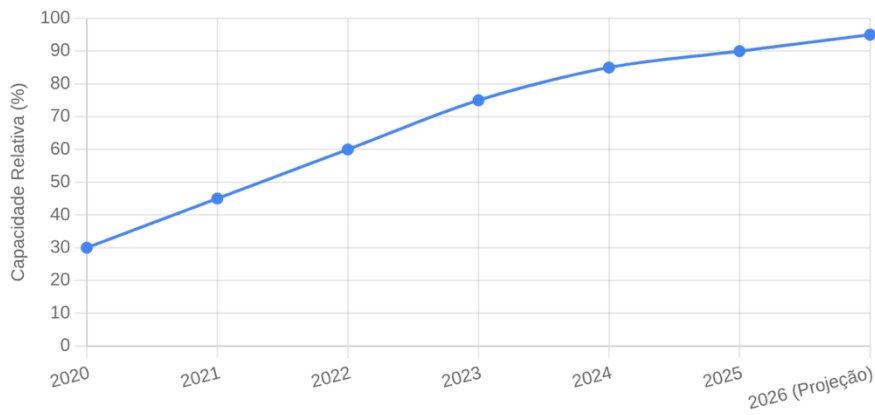
## Recapitulando:

Agentes representam um salto em direção a uma IA mais autônoma e capaz.

LangChain fornece uma estrutura poderosa e flexível para a construção de agentes sofisticados.

O paradigma **LLM + Tools** é a base para a criação de agentes que podem raciocinar e agir no mundo.

O ciclo ReAct (Reasoning + Acting) permite que agentes tomem decisões e executem ações de forma iterativa.



## O Futuro:

### Agentes Mais Autônomos

Capacidade de definir e buscar seus próprios objetivos, com maior independência e iniciativa.

### Melhoria da Memória e Aprendizado

Agentes que aprendem e se adaptam de forma mais eficaz ao longo do tempo, retendo conhecimento e experiências.

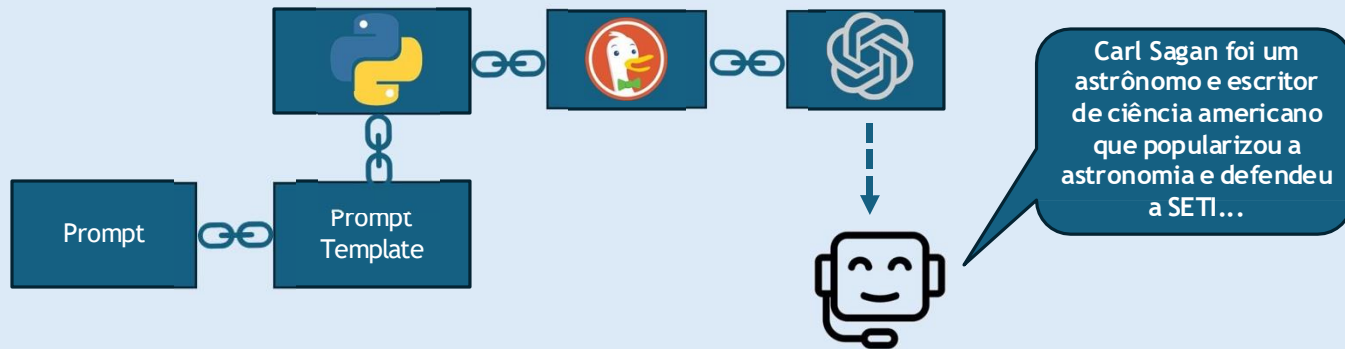
### Ecossistemas de Multi-Agentes

Colaboração complexa entre múltiplos agentes especializados para resolver problemas em larga escala.

### Integração com o Mundo Físico

Robótica e automação impulsionadas por agentes de IA, permitindo interações mais sofisticadas com o ambiente físico.

# Agente de Pesquisa e Resumo





# Agente React Assistente Financeiro

## Pessoal

